Metacat - Bug #474

Metacat in Ecoinfo hang

04/16/2002 04:39 PM - Jing Tao

Status:	Resolved	**Start date:**	04/16/2002
Priority:	Immediate	**Due date:**	
Assignee:	Jing Tao	**% Done:**	0%
Category:	metacat	**Estimated time:**	0.00 hour
Target version:	1.2	**Spent time:**	0.00 hour
Bugzilla-Id:	474		

Description

In our production version of metacat, it often hang in there and need to
restart tomcat. In that time a process of tomcat didn't response either.
But we didn't see any error message.

There are some probable issues to cause this problem
1. DB connection problem. In our code, when open a connection but we didn't
close it at end. I remember when I wrote a sript file to keep sending data to
database, There is a connection problem happend after about 1000 documents
were sent. At that time, I needed to restart tomcat and run script again.

2. Client side sent a request, but it shut down before server sending
response. This cause the problem.

3. Some thread created by metacat never be killed.

4. Unknown other reason.

Please comment it. Thanks.

History

#1 - 05/10/2002 10:28 AM - Jing Tao

Dan tried to access same package from two instances of morpho, and made
metacat hang there.

We tired download one data package from two morpho, the first time worked but
the second time failed.

We use one morpho to download the same package 6 times and it worked fine.

Mutiple thread problem? DB connection problem?

I aked Dan to upload the package to my local machine and I will played with it.

#2 - 05/15/2002 10:55 AM - Jing Tao

We upload the package and play with it couple times. But it did not cause
metacat hang.

I decide to revise db connection code. The idea is create two another classes:
1. DBConnection, it includes Connection, index, status, age, usageCount and
warningMessage attribues. And some methods to get or set these attributes
values. This class will represent a connection object and its information.

2. DBConnectionUtil, the main attributes is "staitic DBConnection []
connectionPool". And the main methods are static getConnection,
returnConncetion, initialConncetionPool, destroyConnectionPool.
This class will be instantiated in MetacatServlet class as an static
attribute. The other class, such as DBQuery, want to use connection, just call
the static getConnection and returnConnection when it finished. So the whole
programm will share one connectionPool. (Now, we didn't do that).

The connection part in MetaCatUtil class will be removed. (I tried to keep it
on MetaCatUtil class rather than write a new class DBConnectionUtil, but

failed. It is better to create an new class).

Suggestions, comments?

#3 - 05/15/2002 11:25 AM - Matt Jones

Sounds like a good idea. I would recommend calling the second class DBConnectionPool rather than DBCOnnectionUtil, because it seems to represent the pool.

Also, be sure we have fields and methods for automatically tracking the state of each connection and the pool for debugging purposes. I think usageCount is a good start.

#4 - 05/15/2002 11:41 AM - Jing Tao

DBConnectionPool will be used as second class's name.

#5 - 05/20/2002 01:32 PM - Jing Tao

Two classes DBConnection and DBConnectionPool were added into cvs. Another test class DBConnectionPoolTester was created too.

DBConnectionPool is a singleton class and this will make sure the whole program share one DBConnection pool. Before a connection checking out, the DBConnection will be checked validation (Age, connection times, and run a simple query getMetaData). A thread will periodically running to check connection validation and print some error or warning too. Moreover, the thread can be configured to run or not.

Here is the test result:
10000 insert query (insert docid into xml_documents table) and 10000 select query were run in diferent thread. If we used MetaCatUtil to manage connection, 6973 records were added and 136 errors (unexpected EOF on client connection) were happend. The error happend means client closed the connection.

If we use the new code, 9653 recods were added. Some failed because duplicated key (I used random()* 1000000 to create docid). Some I don't know why. No error happended during running. If I killed the program, the 10 connections in the pool were closed and caused 10 errors.

May I change the get connection and return connection for whole program? Even there is some bugs in the new code. We need only to fix it the two classes. And the other changes for get connection and return connection for the whole program can still keep same.

Now in Metacat, connection sometime is class field, sometimes is method parameter, sometime is local variable. I would like to make it all local variable. If a method need a connection, just get it from connection pool. It don't need be passed as class field or method parameter.

Suggestions or comments?

#6 - 05/30/2002 09:22 AM - Jing Tao

A test program was applied to JDBC connection. This programm will open a connection and used this connection to execute one insert and one select sql command in a for loop.

But when this program connected to Oracle, it often hang there after executed a mount of sql commands. Here is the result. (the "# of for loop" means after excecute the number of operation, programm hang there).

Time	# of for loop
1.	15844
2.	14832
3.	Couldn't run
4.	6753
5.	Couldn't run
6.	13306
7.	200000 (Successfully finished all operations)
8.	Couldn't run
9.	4110
10.	4282

11. Couldn't run
12. 7009
13. Couldn't run
14. Couldn't run
15. 25448
16. 1000000 (Succeefully finished all operations)
17. 12197
18. 20113
19. Couldn't run
20. 36832

Note: couldn't run means when start the program, it even couldn't finish one operation.

When program hang there, there is no error message. It seem like the metacat did. So I think this is JDBC problem cause the problem.

I will try to increase processes number in init.ora and different JDBC driver to test it.

#7 - 06/03/2002 10:08 AM - Jing Tao

When process inrcrease to 200 in init.oral file, the following is the results:

Time	# of for loop	Reseaon of end
1.	31053	SQL error (Unique constrain violate).
2.	137146	Shut down by me
3.	50232	Hung there
4.	187086	Shut down by me
5.	150000	Running successfully
6.	33546	Hung there
7.	100000	Running successfully

So, increase process number, the result is better.

Here is the testing result for postgresql:

Time	# of for loop	Reseaon of end
1.	50000	Running successfully
2.	50000	Running successfully
3.	121684	Shut down by me

Note: in time 3, I found the speed was very slow after it executed over night. I check the number, it is about 100000. But I am not sure when it stated to slow down.

So postgresql didn't hung there, but speed is very slow after execute a mount of sql commnad.

#8 - 06/03/2002 02:56 PM - Jing Tao

The following is the result by using JDBC 9.0.1 (Process number = 50)

Time	# of loop	Reseaon for end
1.	36596	Hung
2.	7147	Hung
3.	5102	Hung
4.	12763	Hung
5.	6399	Hung

Though in JDBC 9.0.1 readme file, it said it resolved the bug that the JDBC Thin driver hung in some cases during an insert. JDBC 9.0.1 still hung.

So maybe process number is the issue for hung.

#9 - 06/05/2002 07:56 PM - Jing Tao

Here is result for JDBC 9.0.1 when process increased to 200:

Time	# of loop	Reseaon of end
1.	100,000	Success
2.	100,000	Success
3.	100,000	Success
4.	47,393	Hung
5.	200,000	Success
6.	21,556	Hung

#10 - 06/14/2002 10:25 AM - Jing Tao

From above testing, we know JDBC connection always hung in there after it was used for certain amount of times (let us call this number "crital number"). Moreover, we know more process # can increase crital number.

But when process number =200, it was found that sometimes Oracle shut down by itself. I guess current the configuration of our Oracle couldn't support processes = 200. So I change it back to 50. Open_cursors still keeps 100 (I monitor the v$Open_cursor table and found the open_curesor value is almost as same as the active connection number).

My idea is: the minimum critial number is 4110 when JDBC driver is 8.1.6 and process =50. So every DBconnection before checking out should be checked usagecount. If usagecount is greater than 1000, it should be closed and open a new one to replace it. So after a connection check out, it at least can be excuted 3000 times (4110 -1000 = 3110) safelly. This is good for current xml documents. But if document is too big, we still take the risk of hung for inserting it into xml_nodes and xml_index tables. But the possibility of hung is not high (From testing, lots of connection can be excetued more than 10,000 safelly).

Our new code was tested. The package which we used to test is Jornada, it is big. As drescribed before, if two morphos upload this package to metacat at same time and sometimes will cause hung problem. New code was tested by two morpho uploading this package at same time for more than 60 times, hung problem never happened! Dan and I tried to upload this package from three morphos at same time for two times, it worked good too.

I think this problem was fixed in some level and do we need to install it in ecoinfo to see what will happen?

#11 - 10/24/2002 03:01 PM - Jing Tao

Metacat release 1.2 run in ecoinfo for one week. No hang problem happened again. So the bug is fixed.

#12 - 03/27/2013 02:14 PM - Redmine Admin

Original Bugzilla ID was 474